A Appendix

A.1 Implementation Details

For training all models, we initialize the learning rate at 0.001 and reduce it by a factor of 0.1 at 50% and 83% of the total epochs. We utilize a single NVIDIA RTX A6000 GPU for both model training and inference. Training time for each model varies between 7 to 30 GPU hours, depending on the specific model architecture. For adapters and reverters, we start with a learning rate of 0.01, reducing it by a factor of 0.1 after the first epoch. These components are trained in pairs, requiring 1 to 5 GPU hours depending on the specific encoder and decoder architectures.

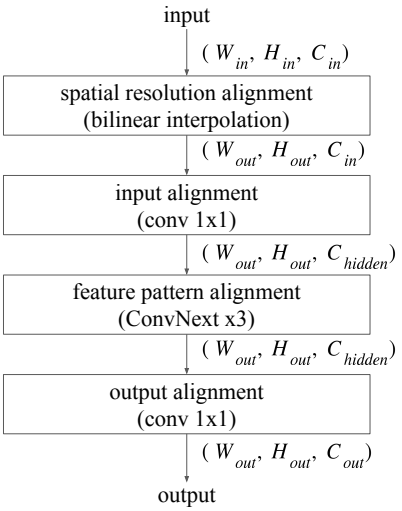

Figure A1: Architecture of adapter and reverter.

Adapter and Reverter's Architecture. We use the same architectures for both adapters and reverters across all CP models, as visualized in Figure A1. The dimension of the broadcasting feature map is set to $(128, 128, 64)$. C_{hidden} is set to be 64. $W_{\text{in}}, H_{\text{in}}, C_{\text{in}}, W_{\text{out}}, H_{\text{out}},$ and C_{out} of adapters and reverters vary according to the feature dimensions of each local model and the broadcasting feature map dimension. For instance, in the task- and model-agnostic setting, Agent 1's feature dimension is $128 \times 128 \times 64$, so we set $(W_{\text{in}}, H_{\text{in}}, C_{\text{in}}) = (W_{\text{out}}, H_{\text{out}}, C_{\text{out}}) = (128, 128, 64)$ for both its adapter and reverter. For Agent 2, with a feature dimension of $64 \times 64 \times 256$, we configure the adapter with $(W_{\text{in}}, H_{\text{in}}, C_{\text{in}}) = (64, 64, 256)$ and $(W_{\text{out}}, H_{\text{out}}, C_{\text{out}}) = (128, 128, 64)$, while the reverter is set with $(W_{\text{in}}, H_{\text{in}}, C_{\text{in}}) = (128, 128, 64)$ and $(W_{\text{out}}, H_{\text{out}}, C_{\text{out}}) = (64, 64, 256)$.

Index	**Agent 1**	**Agent 2**	**Agent 3**	**Agent 4**
Modality	Lidar	Lidar	Camera	Camera
Encoder	PointPillar (Lang et al., 2019)	SECOND (Yan et al., 2018)	EfficientNetB0 (Tan, 2019)	ResNet101 (He et al., 2016)
Encoder Param.(M)	0.87	3.79	56.85	6.88
Index	**Agent 5**	**Agent 6**	**Agent 7**	**Agent 8**
Modality	Camera	Lidar	Camera	Lidar
Encoder	ResNet34 (He et al., 2016)	VoxelNet (Zhou & Tuzel, 2018)	EfficientNetB1 (Tan, 2019)	PointPillar (large) (Lang et al., 2019)
Encoder Param.(M)	6.51	2.13	66.41	1.91
Index	**Agent 9**	**Agent 10**	**Agent 11**	**Agent 12**
Modality	Camera	Lidar	Camera	Lidar
Encoder	ResNet50 (He et al., 2016)	SECOND (large) (Yan et al., 2018)	EfficientNetB2 (Tan, 2019)	VoxelNet (large) (Zhou & Tuzel, 2018)
Encoder Param.(M)	6.88	4.82	71.43	3.18

Table A1: Modality, encoder, and encoder parameters (M) of each heterogeneous model in the 3D object detection setting.

3D object detection setting. Under the experiments on 3D object detection task, we prepared 12 heterogeneous models. Table A1 displays the Modality, Encoder, and Encoder Parameters (M) information of each of the 12 heterogeneous models. For model 7, 9, and 11, we enlarge the encoders by increasing the size of hidden layers. For all heterogeneous models, we choose pyramid fusion layers proposed by Lu et al. (2024) to be the fusion module and three 1×1 convolutional layers for classification, regression, and direction, respectively.

A.2 Architectural Comparison between Existing Frameworks

Figure A2 illustrated various frameworks that address heterogeneous CP. Late fusion simply combines agent outputs through post-processing. Calibrator (Xu et al., 2023) enhances this approach by using calibrators to address domain gaps between heterogeneous agent outputs. End-to-end training, while effective, lacks scalability due to its requirement of re-training all agents' models. It also compromises security and task flexibility by shared fusion models and decoders. HEAL (Lu et al., 2024) improves upon this by fixing decoders and fusion models, re-training only the encoders, reducing training resources but still facing scalability issues due to the computational cost of encoder retraining as well as the security issue due to the shared fusion models and decoders. Our proposed framework, STAMP, introduces a novel approach using lightweight adapter and reverter pairs to align feature maps for collaboration. The lightweight nature of these components ensures scalability, while the maintenance of local fusion and decoders ensures both security and task agnosticism. This design effectively addresses the limitations of previous methods.

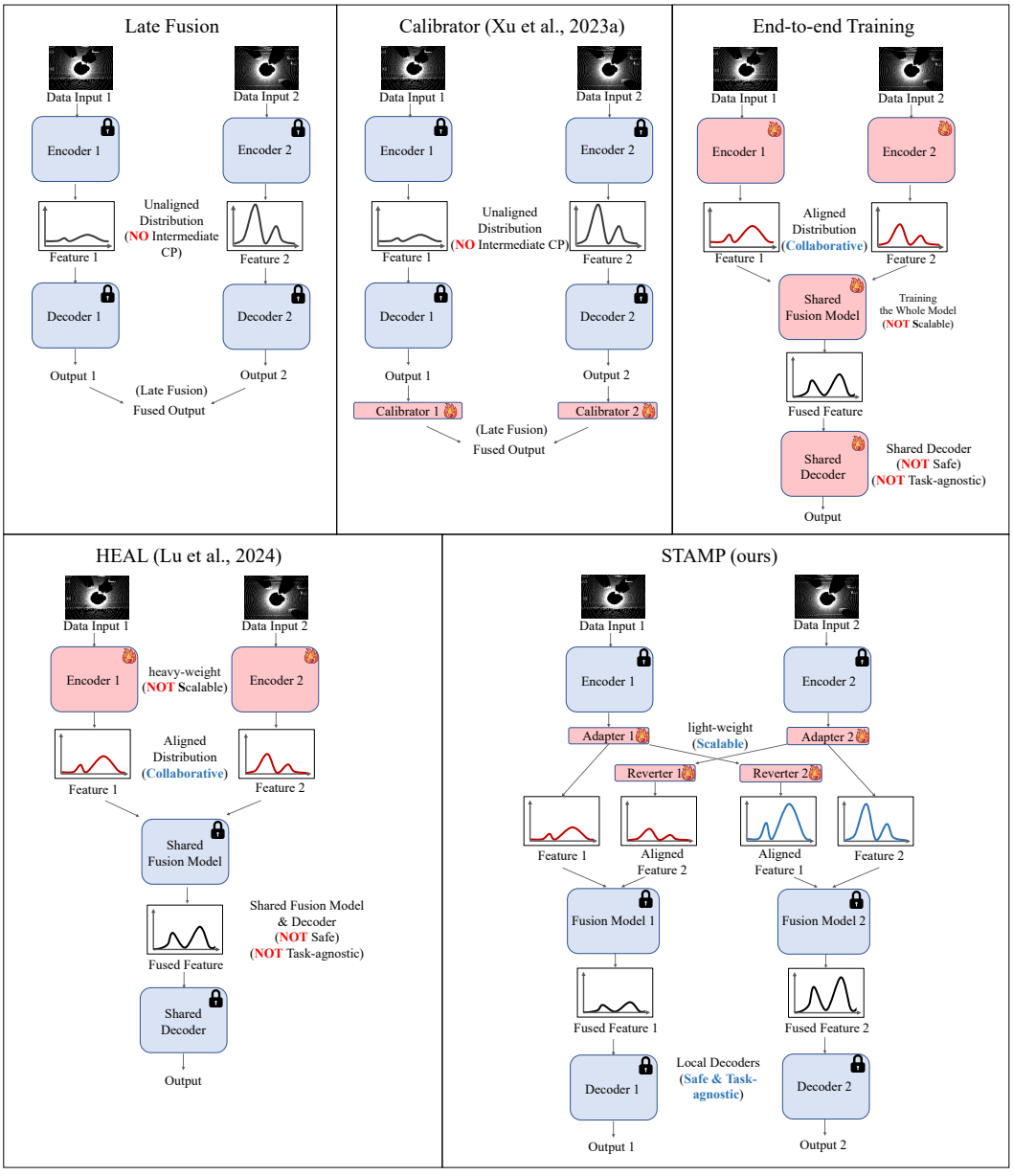

Figure A2: Architectural comparison of collaborative perception frameworks: existing approaches versus our proposed STAMP method. Blue boxes represent models with fixed parameters, while red boxes indicate models whose parameters are trained during the collaboration process.

A.3 Multi-group and Multi-model Collaborations System

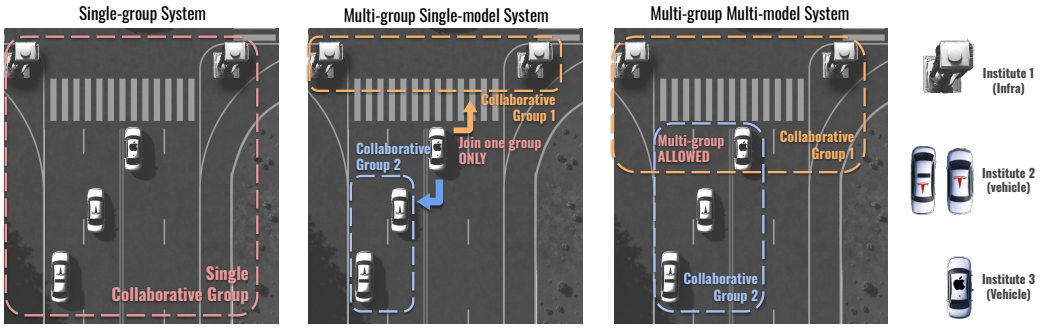

Figure A3: Comparison of collaborative perception systems: (Left) Single-group system where all agents collaborate within one group. (Middle) Multi-group single-model system allowing agents to join only one of multiple collaboration groups. (Right) Multi-group multi-model system enabling agents to participate in multiple collaboration groups simultaneously. The figure illustrates how different system architectures impact agent interactions and group formations in autonomous driving scenarios.

In our experimental findings, we observed a bottleneck effect in CP systems, where the overall system performance is constrained by the capabilities of the weakest agent. This limitation underscores the need for more selective collaboration, leading us to introduce the concept of a **Collaboration Group** - a set of agents that collaborate under specific criteria. These criteria are essential for maintaining the quality and integrity of CP, admitting agents that meet predefined standards while excluding those with inferior models, potential malicious intent, or incompatible alignments. As illustrated in Figure A3, we can distinguish between three collaborative system types:

- Single-group systems, where agents either operate independently or are compelled to collaborate with all others, are susceptible to performance bottlenecks caused by inferior agents and vulnerabilities introduced by malicious attackers.
- Multi-group single-model systems, allowing multiple collaboration groups but restricting agents to a single group because each agent can only equip a single model.
- Multi-group multi-model systems, enabling agents to join multiple groups if they meet the predefined standards.

The multi-group structure offers significant advantages over traditional single-group systems. It enhances agents' potential for diverse collaborations, consequently improving overall performance. This approach mitigates the bottleneck effect by allowing high-performing agents to maintain efficiency within groups of similar capability while potentially assisting less capable agents in other groups. Furthermore, it enhances system flexibility, enabling dynamic group formation based on specific task requirements or environmental conditions.

However, implementing such a multi-group system poses challenges for existing heterogeneous collaborative pipelines. End-to-end training approaches require simultaneous training of all models, conflicting with the concept of distinct collaboration groups. Methods like those proposed by Lu et al. (2024) require separate encoders for each group, becoming impractical as the number of groups increases due to computational and memory constraints.

Our proposed STAMP framework effectively addresses these limitations, offering a scalable solution for multi-group CP. The key innovation lies in its lightweight adapter and reverter pair (approximately 1MB) required for each collaboration group an agent joins. This efficient design enables agents to equip multiple adapter-reverter pairs, facilitating seamless participation in various groups without significant computational overhead. The minimal memory footprint ensures scalability, even as agents join numerous collaboration groups, making STAMP particularly well-suited for multi-group and multi-model collaboration systems.

A.4 More Visualization Results

Figure A4 and A5 illustrate more feature map and result visualizations before and after collaborative feature alignment (CFA). Prior to CFA, agents' feature maps exhibit disparate representations. For instance, in Figure A4, the pre-fusion feature maps of agents 1, 3, and 4 appear entirely black, indicating a significantly lower scale compared to agent 2's feature map. This discrepancy leads to instability in feature fusion. Post-CFA, the features are aligned to the same domain, resulting in more coherent fusion and accurate inference outputs.

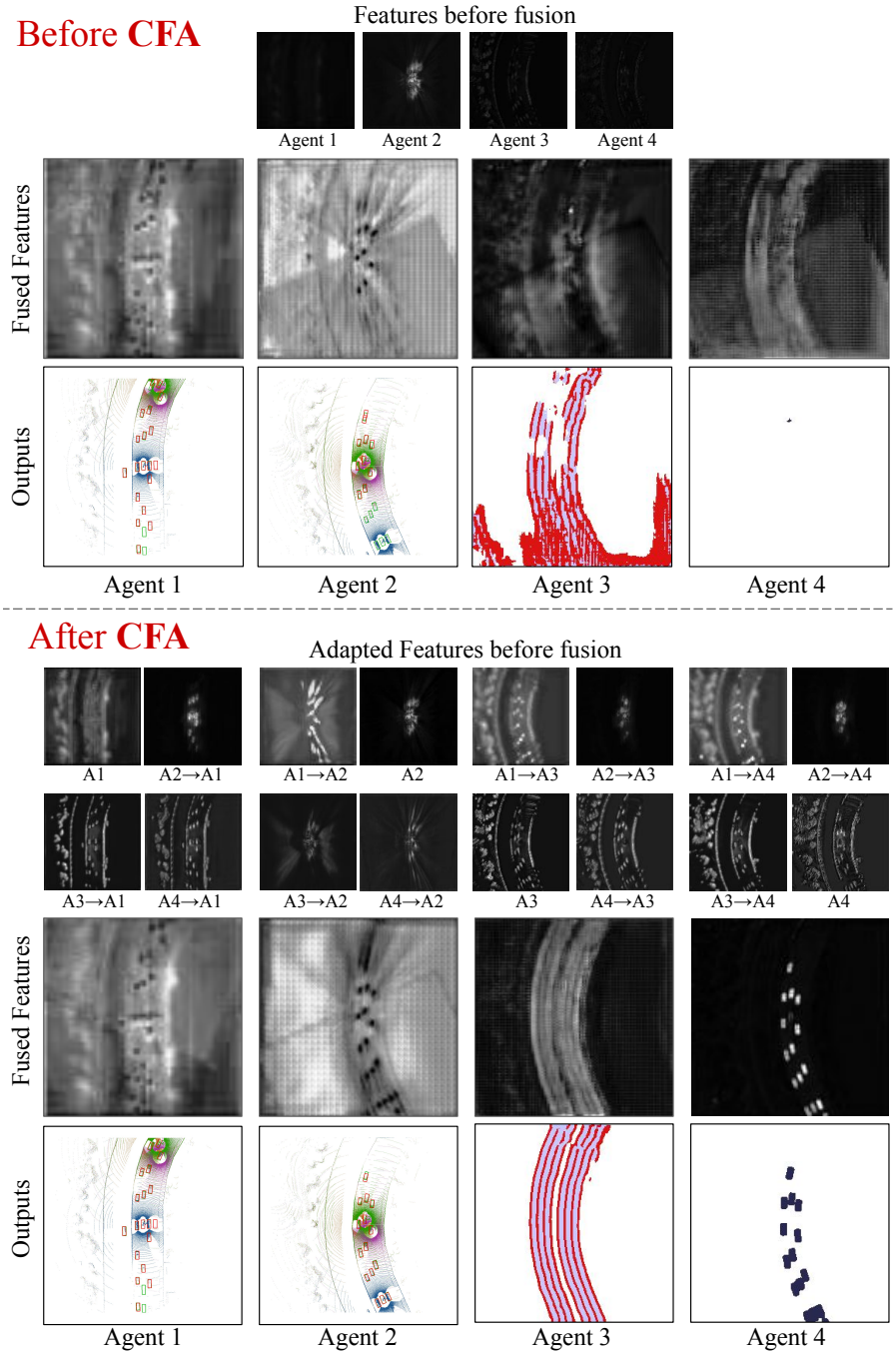

Figure A4: Visualization of feature maps and inference results before and after Collaborative Feature Alignment (CFA) in a three-agent scene. $A_i \rightarrow A_j$ denotes the feature map aligned from agent i's domain to agent j's domain, also represented as F_{ij}.

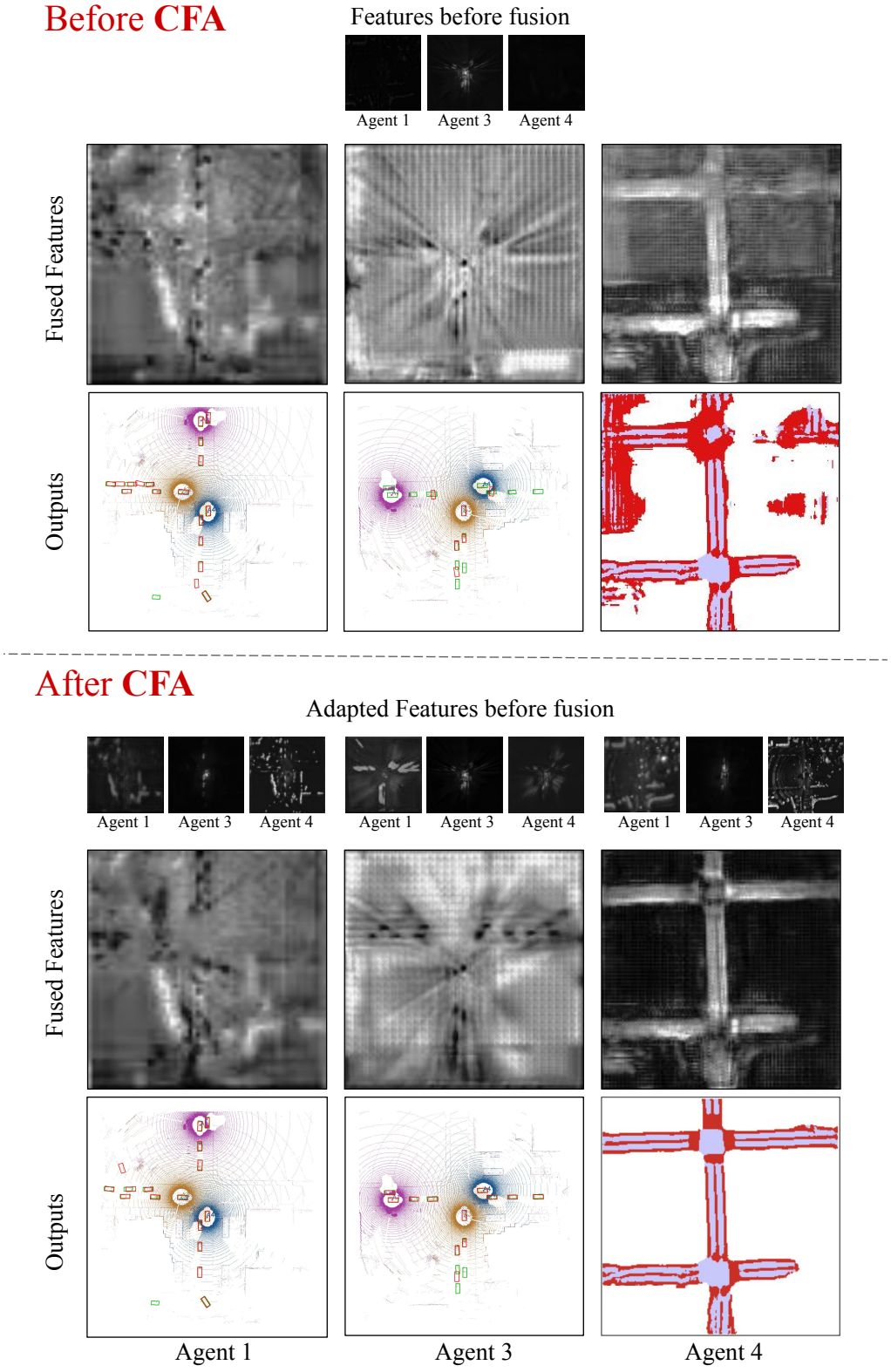

Figure A5: Visualization of feature maps and inference results before and after Collaborative Feature Alignment (CFA) in a four-agent scene. $A_i \rightarrow A_j$ denotes the feature map aligned from agent i's domain to agent j's domain, also represented as F_{ij}.

REFERENCES

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 1

Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12697–12705, 2019. 1

Yifan Lu, Yue Hu, Yiqi Zhong, Dequan Wang, Siheng Chen, and Yanfeng Wang. An extensible framework for open heterogeneous collaborative perception. *arXiv preprint arXiv:2401.13964*, 2024. 1, 2, 3

Mingxing Tan. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019. 1

Runsheng Xu, Weizhe Chen, Hao Xiang, Xin Xia, Lantao Liu, and Jiaqi Ma. Model-agnostic multi-agent perception framework. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1471–1478. IEEE, 2023. 2

Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1

Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4490–4499, 2018. 1